|  |  |
| --- | --- |
| **NAME:** | Shubham Golwal |
| **UID NO.** | 2020300015 |
| **Experiment No.** | 03 |

|  |  |
| --- | --- |
| **AIM:** | Vertical fragmentation |
| **Program 1** | |
| **PROBLEM STATEMENT :** | Write the scenario of any application in distributed database and consider, one relation from that and fragment that relation into vertical fragmentation and execute the queries on that fragment |
| **THEORY :** | **What is Fragmentation?**  Fragmentation in ADBMS (Advanced Database Management Systems) refers to the process of dividing a large database into smaller and more manageable parts, called fragments. This is done to improve the performance, scalability, and availability of the database by distributing the data across multiple servers, disks, or storage devices. This can also help in reducing the size of individual fragments and improve the access time for specific data subsets, making it easier to manage and maintain the database.  There are two main types of fragmentation in ADBMS:   1. Horizontal Fragmentation 2. Vertical Fragmentation   **Vertical Fragmentation**   * Vertical fragmentation is a database design technique used to divide a large table into smaller tables based on their columns. This technique helps to improve query performance by reducing the amount of data that needs to be scanned by the database engine. * By breaking a large table into smaller tables, the database engine can scan a smaller set of data to answer a query, which can result in faster query performance. * Vertical fragmentation can also improve data management by separating related columns into smaller, more manageable tables. * If a particular column is frequently queried, it can be placed in a separate table to improve performance. Alternatively, if a column is rarely used, it can be placed in a separate table to reduce the size of the main table and improve performance.   **Advantages of fragmentation:**  Before we discuss fragmentation in detail, we list four reasons for fragmenting a relation  **Usage**  In general, applications work with views rather than entire relations. Therefore, for data distribution, it seems appropriate to work with subsets of relation as the unit of distribution.  **Efficiency**  Data is stored close to where it is most frequently used. In addition, data that is ,not needed by’ local applications is not stored.  **Parallelism**  With fragments as the unit of distribution, a transaction can be divided into several sub queries that operate on fragments. This should increase the degree of concurrency, or parallelism, in the system, thereby allowing transactions that can do so safely to execute in parallel.  **Security**  Data not required by local applications is not stored, and consequently not available to unauthorized users.  **Disadvantages of fragmentation**  Fragmentation has two primary disadvantages, which we have mentioned previously:  **Performance**  The performance of global application that requires data from several fragments located at different sites may be slower.  **Integrity**  Integrity control may be more difficult if data and functional dependencies are fragmented and located at different sites. |
| **OUTPUT:**  Creation and data insertion in bank details table :  Vertical Fragmentation of bank details table  **Creation and insertion of data in cust table:**  **Creation and insertion of data in cust\_loan table :**  **Creation and insertion of data in transaction table:**    **Observations on Fragmented Tables:**   1. Retrieve the details of all customers who have made a deposit of more than or equal to 3000      1. Retrieve the total number of transactions made via each mode of transaction      1. Retrieve the details of all customers with due loan amount greater than 3000      1. Retrieve the details of all transactions made after December 2022      1. Retrieve the details of all customers who made deposit through cheque     **Q: For the vertical fragments check the correctness rules**  **Completeness :** If relation R is decomposed into fragments R1,R2,....Rn each data item that can be found in R can also be found in one or more Ri’s.  **Ans:** From the above query executions we can say that records present in emp can be found either in cust or in cust\_loan.  **Reconstruction:** If relation R is decomposed into fragments R1,R2,...Rn , it should be possible to define relational operator delta such that R=delta(Ri) for all Ri belongs to Fr.  **Ans:** When we Join both the tables, we are able to get the original table back, so the reconstruction property stands. | |
| **CONCLUSION:** Thus, we have performed vertical fragmentation on the database and checked the  correctness rules for the same. And after successfully completing this experiment, I learnt that:   1. Vertical fragmentation is a useful technique for improving query performance and data management in sql databases. 2. Vertical fragmentation helps to reduce query processing time. 3. Vertical fragmentation makes easier to update and maintain the data, as well as improve data security by limiting access to specific columns. 4. Vertical fragmentation can increase the flexibility of the database design | |